Accessing DB2 UDB Data

Presented by DB2 Developer Domain

`http://www7b.software.ibm.com/dmdd/`

_____

Table of Contents

If you're viewing this document online, you can click any of the topics below to link directly to that section.

Section 1. Introduction

What this tutorial is about

This tutorial is the third in a series of six tutorials that can help you prepare for the DB2 UDB V8.1 Family Fundamentals Certification (Exam 700). The material in this tutorial primarily covers the objectives in Section 3 of the test, which is entitled "Accessing DB2 UDB Data." You can view these objectives at *http://www.ibm.com/certify/tests/obj700.shtml*.

DB2 installation is not covered in this tutorial. If you haven't already done so, we strongly recommend that you download and install a copy of *IBM DB2 Universal Database*, Enterprise Server Edition. Installing DB2 will help you understand many of the concepts that are tested on the DB2 UDB V8.1 Family Fundamentals Certification exam. The installation process is documented in the Quick Beginnings books, which can be found at the *DB2 Technical Support* Web site under the Technical Information heading.

After you've installed the DB2 product, you will want to get a database up and running as quickly as possible. This tutorial introduces you to the objects that make up a DB2 database, and to the factors that affect how the database is created. After a brief introduction to DB2 objects, we'll examine the different tools to create, access, and manipulate DB2 objects.

By the end of this tutorial, you should be able to create a DB2 database on your own, catalog it for use by other users, and examine and manipulate the objects within that database.

About the author

George Baklarz, B.Math, M.Sc (Comp Sci), is the manager of the DB2 Worldwide Pre-sales Support Group. He has more than nineteen years of experience with DB2 and has co-authored the *DB2 UDB Version 8.1 Database Administration Certification Guide* (Prentice-Hall, 2003).

In his spare time he teaches database theory at the University of Guelph and presents at a variety of conferences, including the International DB2 Users Group.

You can reach George when he's not traveling at *baklarz@yahoo.com*.

Section 2. What makes up a DB2 database?

Logical, physical, and performance features of a database

A DB2 database is actually made up of a collection of objects. From the user's perspective, a database is a collection of tables that are (hopefully) related in some way.

From the perspective of a database administrator (DBA -- that's you), it's a little more complicated than that. The actual database contains many of the following physical and logical objects:

- Tables, views, indexes, schemas
- Locks, triggers, stored procedures, packages
- Buffer pools, log files, tablespaces

Some of these objects, like tables or views, help determine how the data is organized. Other objects, like tablespaces, refer to the physical implementation of the database. Finally, some objects, like buffer pools and other memory objects, only deal with how the database performance is managed.

Rather than dwell on all possible combinations of parameters and objects, the DBA should first concentrate on the physical implementation of the database. How do you create a database and allocate the disk storage required for it? To properly answer that question, you need to know about the basic objects in the database and how they get mapped to physical disk storage.

The DB2 storage model

DB2 has both a logical and physical storage model to handle data. The actual data that users deal with is found in *tables*. While tables may be made up of columns and rows, the user has no knowledge of the physical representation of the data. This fact is sometimes referred to as the *physical independence* of the data.

The tables themselves are placed into *tablespaces*. A tablespace is used as a layer between the database and the container objects that hold the actual table data. A tablespace can contain more than one table.

A *container* is a physical storage device. It can be identified by a directory name, a device name, or a file name. A container is assigned to a tablespace. A tablespace can span many containers, which means that you can get around operating system limitations that may limit the amount of data that one container can have. The relationship between all of these objects is illustrated in the figure below.

Tablespace

Tables

Containers
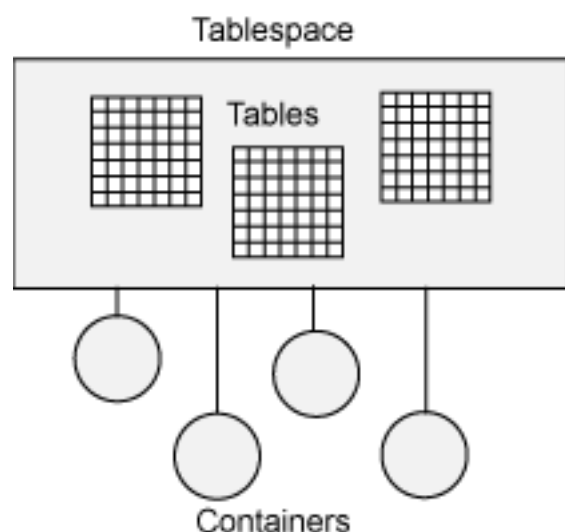
Although a table is the basic object that is placed into a tablespace, a DBA must be aware of additional objects within the DB2 system and how they are mapped to a tablespace.

Tables, indexes, long fields, and tablespaces

Tables, indexes and long fields (sometimes called binary large objects or BLOBs) are objects that are created within a DB2 database. These objects get mapped to a tablespace that is itself mapped to physical disk storage.

A table is an unordered set of data records. It consists of columns and rows that are generally known as *records*. Tables can be either permanent (base) tables, temporary (declared) tables, or temporary (derived) tables. From a DBA perspective, space is allocated for each one of these table objects, but in different tablespaces.

An *index* is a physical object that is associated with a single table. Indexes are used to enforce uniqueness in a table (that is, to make sure that there are no duplicate values) and to improve performance when retrieving information. You don't need indexes to run your SQL (Structured Query Language) statements; however, your users will appreciate your foresight in creating a few of them to speed up query processing!

A *long field* (or *BLOB*) is a type of data found within a table. This data type is typically made up of unstructured data (an image, a document, an audio file) and usually contains a significant amount of information. Storing this type of data within a table would lead to excessive overhead when deleting, inserting, and manipulating these objects. Instead of storing them directly in the row of the table, a pointer is stored instead that links to a spot in a Large tablespace (previously known as a Long Field tablespace). DBAs need to be aware of this datatype so they can create the appropriate tablespaces to contain these objects.

Armed with the knowledge of these different object types, you are now ready to determine the type of space that you need to allocate.

DMS and SMS tablespaces

Tablespaces are the logical layer between the database and the tables stored in that database. Tablespaces are created within a database and tables are created within tablespaces. DB2 supports two kinds of tablespaces:

- **System-Managed Space (SMS):** Here, the operating system's file system manager allocates and manages the space, where the table is the default tablespace type.
- **Database-Managed Space (DMS):** Here, the database manager controls the storage space. This tablespace is, essentially, an implementation of a special-purpose file system designed to best meet the needs of the database manager.

SMS tablespaces require very little maintenance. However, SMS tablespaces offer fewer optimization options and may not perform as well as DMS tablespaces.

So, which tablespace design should you choose?

DMS vs. SMS

Although the following table is not exhaustive, it does contain some things for you to consider when deciding between DMS and SMS tablespaces.

Feature	SMS	DMS
Striping?	Yes	Yes
Object management	Operating system	DB2
Space allocation	Grows/shrinks on demand	Preallocated; size can shrink and grow but requires DBA intervention.
Ease of administration	Best; little or no tuning required	Good, but some tuning required (e.g.. EXTENTSIZE PREFETCHSIZE)
Performance	Very good	Best; can achieve up to 5 to 10% advantage with raw containers

Summary

We've covered a lot of ground in the section. Let's summarize what we've learn about DB2 databases.

- A database is a collection of objects, which includes tables, indexes, views, and long objects.
- These objects are stored within tablespaces, which in turn are made up of containers.
- Tablespaces can be either managed by the operating system (SMS) or by DB2 (DMS).
- You'll decide which type of tablespace to use based mostly on performance and maintenance factors.

Now that you are an expert on the different types of tablespaces, it's time to create your first database. The next section will show you how.

Section 3. Creating your first database

First Steps

As part of the DB2 installation process, the First Steps panel is displayed allowing the user to generate a number of a sample databases to work with:
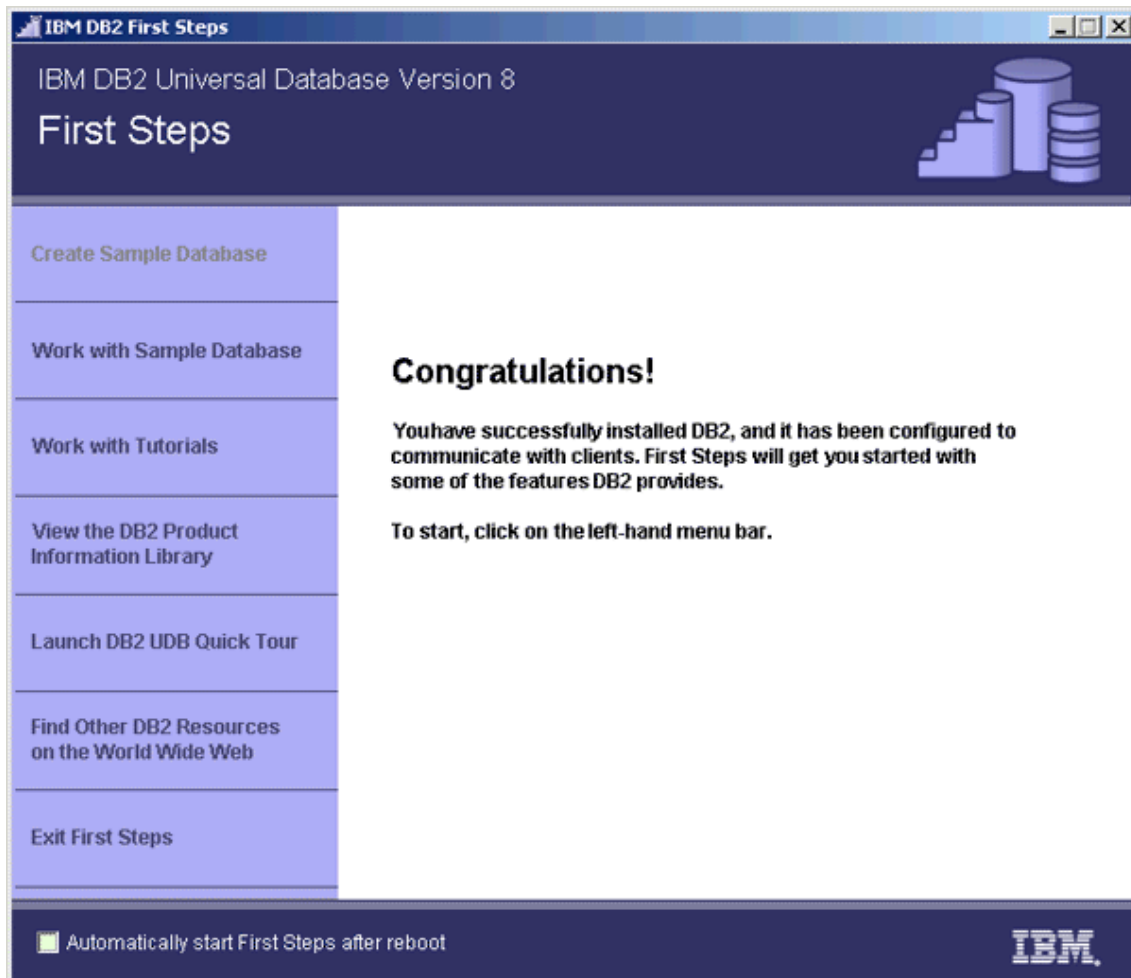

Most users will want to create the SAMPLE database and use that to explore the features of DB2. This panel can be invoked by selecting First Steps from within the Setup Tools folder in the DB2 Program group (in Windows environments). In addition, issuing the command db2sampl from a command-line prompt will also generate the SAMPLE database.

For more information on the First Steps tool, check out the *first tutorial in this series*.

Let's move on to creating a DB2 database without a GUI!

_____

My first real database

Creating a DB2 database from a command line is relatively simple. To create a database, you must invoke the DB2 Command Line Processor (CLP). This can be accomplished by either selecting Command Line Processor from the Command Line Tools folder in the DB2 Program Group (see the figure below), or executing the command `db2cmd db2` from an operating system command line. (For more information on the Command Line Processor, check out the *first tutorial in this series*.)
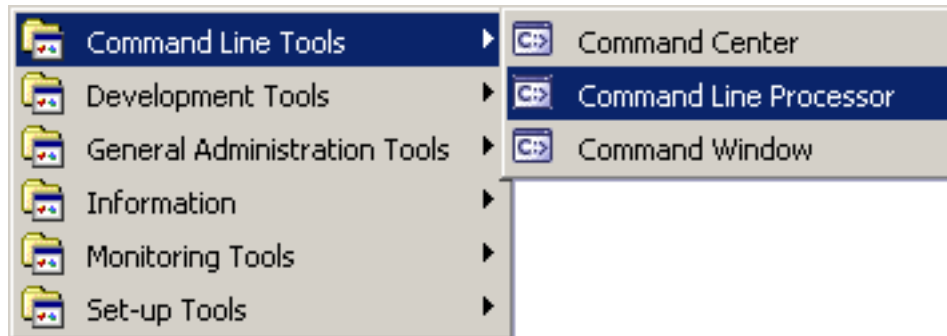
The syntax for creating a DB2 database is as follows:

```
-CREATE DATABASE MY1STDB
```

"That's it?" you ask? That's it! The only element that is required as part of a CREATE DATABASE command is the name of the database. The rules for a database name are:

- The database name can consist of the following characters: a-z, A-Z, 0-9, @, #, and $.

- The first character in the name must be an alphabetic character, @, #, or $; it cannot be a number or the letter sequences SYS, DBM, or IBM.

- A database name or database alias is a unique character string containing from one to eight letters, numbers, or keyboard characters from the set described above.

Of course, there are many more options that are available to you; you don't have to only enter a name. Let's examine what actually happened as a result of this command.

What did DB2 create?

When you issued the CREATE DATABASE command, DB2 created a number of files. These files include log files, configuration information, history files, and three tablespaces. These tablespaces are:

- SYSCATSPACE: This is where the DB2 system catalog is kept that tracks all of the metadata associated with DB2 objects.

- TEMPSPACE1: A temporary work area where DB2 can place intermediate results.
- USERSPACE1: A place where all user objects (tables, indexes) reside by default.

All of these files are placed into the DB2 directory found on your default drive. The default drive is typically the volume on which you installed the DB2 product.

For simple applications, this default configuration may be sufficient for your needs. However, you may want to change the location of your database files, or change the way DB2 manages these objects. In the next panel, we'll explore the CREATE DATABASE command in more detail.

The CREATE DATABASE command

The full syntax of the DB2 CREATE DATABASE command can be found in the DB2 Command Reference, but the following diagram illustrates the majority of options that a DBA would be interested in.

```
>>-CREATE--+-DATABASE-+--database-name------------------------->
           '-DB-------'
>--+----------------------------+----------------------------->
   '-| Create Database options |-'
Create Database options
|--+-------------+--+---------------------+---------------->
   '-ON--+-path--+-'  '-ALIAS--database-alias-'
         '-drive-'
>--+-------------------------------------------+------------->
   '-USING CODESET--codeset--TERRITORY--territory-'
>--+--------------------------------+----------------------->
   |              .-SYSTEM--------. |
   '-COLLATE USING--+-COMPATIBILITY-+-'
                    +-IDENTITY------+
                    '-NLSCHAR-------'
>--+---------------------------------+------------------->
   '-CATALOG TABLESPACE--| tblspace-defn |-'
>--+---------------------------------+------------------->
   '-USER TABLESPACE--| tblspace-defn |-'
>--+------------------------------------+--------------->
   '-TEMPORARY TABLESPACE--| tblspace-defn |-'
tblspace-defn
|--MANAGED BY------------------------------------------->
                    .-,----------------.
                    V                  |
>--+-SYSTEM USING--(----'container-string'-+--)-------------------------------+-->
   |                       .-,------------------------------------------.     |
   |                       V                                            |     |
   '-DATABASE USING--(----+-FILE---+--'container-string'--number-of-pages-+--)-'
                          '-DEVICE-'
>--+-------------------------------+----------------------->
   '-EXTENTSIZE--number-of-pages-'
>--+--------------------------------+---------------------->
   '-PREFETCHSIZE--number-of-pages-'
```

In the following panels, you'll learn what these various options are and how you would

use them.

Database location

One of the parameters of the CREATE DATABASE command is the `ON path/drive` option. This option tells DB2 where you want to create the database.

On UNIX-based systems, this option specifies the path on which to create the database. If a path is not specified, the database is created on the default database path specified in the Database Manager Configuration (DBM CFG) file (in the dftdbpath parameter).

On Windows operating systems, this option specifies the letter of the drive on which to create the database.

For example, the following CREATE DATABASE command places the database on the D: drive on a Windows operating system:

```
CREATE DATABASE MYDB ON D:
```

Code pages and collating sequences

A character code page is associated with all DB2 character data types (CHAR, VARCHAR, CLOB, DBCLOB). A code page can be thought of as a reference table used to convert alphanumeric data to the binary data that is stored in the database. A DB2 database can only use a single code page. The code page is established during the CREATE DATABASE command using the options `CODESET` and `TERRITORY`. The code page can use a single byte to represent an alphanumeric character (a single byte can represent 256 unique elements) or multiple bytes.

Languages like English contain relatively few unique characters; therefore, a single-byte code page is sufficient to store data. Languages like Japanese require more than 256 elements to represent all of their unique characters; therefore, a multibyte code page (usually a double-byte code page) is required.

By default, the collating sequence of a database is defined according to the codeset used in the CREATE DATABASE command. If you specify the option `COLLATE USING SYSTEM`, the data values are compared based on the `TERRITORY` specified for the database. If the option `COLLATE USING IDENTITY` is used, all values are compared using their binary representation in a byte-by-byte manner.

The DB2 Administration Guide lists the various code pages that are available when creating a database. In most instances, a DBA would let this value default to the same code page as the operating system that the database will run on.

Tablespace definitions

Each of our three tablespaces (SYSCATSPACE, TEMPSPACE1, USERSPACE1) are
created automatically in the default directory (ON keyword) unless you specify their
location. For each tablespace, the DBA can specify the characteristics of the file
system that the tablespace should to use.

The three tablespaces are defined using the following syntax:

```
>--+-------------------------------------+------------------->
   '-CATALOG TABLESPACE--| tblspace-defn |-'

>--+-------------------------------------+--------------------->
   '-USER TABLESPACE--| tblspace-defn |-'

>--+-------------------------------------+------------------->
   '-TEMPORARY TABLESPACE--| tblspace-defn |-'
```

If any of these keywords are omitted, DB2 will use the default values to generate the
tablespaces. The tablespace definition follows these options and has the following
syntax:

```
|--MANAGED BY----------------------------------------------->


                      .-,----------------.
                      V                  |
>--+-SYSTEM USING--(----'container-string'-+--)----------------------------------+-->
   |                      .-,---------------------------------------------.     |
   |                      V                                             |     |
   '-DATABASE USING--(----+-FILE---+--'container-string'--number-of-pages-+--)-'
                          '-DEVICE-'

>--+-------------------------+------------------------------->
   '-EXTENTSIZE--number-of-pages-'

>--+-------------------------+------------------------------->
   '-PREFETCHSIZE--number-of-pages-'
```

Let's look at this syntax in detail. The MANAGED BY option tells DB2 to generate these
tablespaces and determine how the space will be managed. SMS tablespaces use the
SYSTEM USING keyword, as follows:

```
SYSTEM USING ('container string')
```

For an SMS tablespace, the *container string* identifies one or more containers that will
belong to the tablespace and into which the tablespace's data will be stored. Each
container string can be an absolute or relative directory name. The directory name, if
not absolute, is relative to the database directory. If any component of the directory
name does not exist, it is created by the database manager. The format of the
container string is dependent on the operating system.

DMS tablespaces are defined with the DATABASE USING keyword:

```
DATABASE USING ( FILE/DEVICE 'container string' number of pages )
```

For a DMS tablespace, the container string identifies one or more containers that will belong to the tablespace and into which the tablespace's data will be stored. The type of the container (either FILE or DEVICE) and its size (in PAGESIZE pages) are specified. The size can also be specified as an integer value followed by K (for kilobytes), M (for megabytes) or G (for gigabytes). You can specify a mixture of FILE and DEVICE containers.

For a FILE container, the container string must be an absolute or relative file name. The file name, if not absolute, is relative to the database directory. If any component of the directory name does not exist, it is created by the database manager. If the file does not exist, it will be created and initialized to the specified size by the database manager. For a DEVICE container, the container string must be a device name and the device must already exist.

One important note: All containers must be unique across all databases; a container can belong to only one tablespace.

```
EXTENTSIZE number of pages
```

EXTENSIZE specifies the number of PAGESIZE pages that the database will write to a container before skipping to the next container. The EXTENSIZE value can also be specified as an integer value followed by K, M, or G. The database manager cycles repeatedly through the containers as data is stored.

```
PREFETCHSIZE number of pages
```

PREFETCHSIZE specifies the number of PAGESIZE pages that will be read from the tablespace when data prefetching is being performed. The prefetch size value can also be specified as an integer value followed by K, M, or G.

Prefetching reads in data needed by a query prior to it being referenced by the query, so that the query need not wait for the underlying system to perform I/O operations.

Sample CREATE DATABASE command

The following is an example of a CREATE DATABASE command that uses many of the options that we discussed in previous panels.

```
( 1) CREATE DATABASE MY1STDB
( 2)  DFT_EXTENT_SZ 4
( 3)  CATALOG TABLESPACE MANAGED BY DATABASE USING
( 4)   (FILE 'C:\CAT\CATALOG.DAT' 2000, FILE 'D:\CAT\CATALOG.DAT' 2000)
```

```
( 5)    EXTENTSIZE 8
( 6)   PREFETCHSIZE 16
( 7) TEMPORARY TABLESPACE MANAGED BY SYSTEM USING
( 8)  ('C:\TEMPTS','D:\TEMPTS')
( 9) USER TABLESPACE MANAGED BY DATABASE USING
(10)   (FILE 'C:\TS\USERTS.DAT' 121)
(11)   EXTENTSIZE 24
(12)   PREFETCHSIZE 48
```

Let's look at each line in more detail:

1. `CREATE DATABASE`: This statement defines the name of the database we are creating.

2. `DFT_EXTENT_SZ 4`: This parameter tells DB2 that the default extent size is 4 pages unless explicitly stated otherwise.

3. `CATALOG TABLEPSACE MANAGED BY DATABASE USING`: The DB2 catalog space will be managed by the database.

4. `FILE 'C:\....'`: The location of the tablespace will be split across two files, each with 2,000 pages of space.

5. `EXTENTSIZE 8`: The `EXTENTSIZE` will be 8 pages.

6. `PREFETCHSIZE 16`: During query processing, 16 pages will be read in at once.

7. `TEMPORARY TABLESPACE MANAGED BY SYSTEM USING`: The temporary space used by DB2 will be handled by the operating system.

8. `'C:\TEMPTS' ...`: The temporary space will be split across two files whose size is automatically adjusted during DB2 execution.

9. `USER TABLESPACE MANAGED BY DATABASE USING`: The user space (where the real tables are placed) will be managed by DB2 directly.

10. `FILE 'C:\TS\...'`: There is only one container for this space and it consists of 121 pages.

11. `EXTENTSIZE 24`: The `EXTENTSIZE` for the `USER` tablespace will be 24 pages.

12. `PREFETCHSIZE 48`: Queries will prefetch 48 pages at once.

Summary

This section gave you some background on how to create a DB2 database. In most

cases, the default values of the CREATE DATABASE command will give you a database that you can use for development and testing.

Once you decide to place a database into production, you'll need to put more effort into the data placement and the tablespace definitions used by DB2. While this might take more planning, the end result will be a database that is easier to manage with potentially better performance.

Section 4. Cataloging your DB2 Database

Why catalog a database?

DB2 automatically catalogs databases when they are created. It catalogs an entry for the database in the local database directory and another entry in the system database directory. If the database is created from a remote client (or for a client that is executing from a different instance on the same machine), an entry is also made in the system database directory at the client instance.

So, why does a database have to be cataloged? Without this information, an application can't connect to a database! DB2 has multiple directories that are used to access databases. These directories allow DB2 to find databases known to it whether they are on the local system or a remote system. The system database directory contains a list and pointer indication where each of the known databases can be found. The node directory contains information relating to how and where remote systems or instances can be found. To put an entry into any of these directories, a CATALOG command is used. To remove an entry, the UNCATALOG command is used.

The CATALOG command

The CATALOG command is diagrammed below.

```
>>-CATALOG--+-DATABASE-+--database-name--+-----------+---------->
            '-DB-------'                 '-AS--alias-'

>--+----------------+------------------------------------------->
   +-ON--+-path--+-----+
   |     '-drive-'     |
   '-AT NODE--nodename-'

>--+------------------------------------------------------------+-->
   |                   .-SERVER----------------------------------. |
   '-AUTHENTICATION--+-CLIENT----------------------------------+-'
                     +-SERVER_ENCRYPT--------------------------+
                     '-KERBEROS TARGET PRINCIPAL--principalname-'

>--+-----------------------+------------------------------->
   '-WITH--"comment-string"-'
```

Cataloging a database is relatively straightforward. Normally this step is not required when you have created a database. However, you may need to do so if you have previously uncataloged the database, if you want to set up an ALIAS (alternate name) for this database, or if you need to access this database from a client.

Cataloging at the client

A user who needs to connect to a DB2 database should catalog the database on a local workstation. In order to do this, the user would use the CATALOG command or the DB2 Configuration Assistant (CA). The CA lets you maintain a list of databases to which your applications can connect. It catalogs nodes and databases while shielding the user from the inherent complexities of these tasks. (For more information on the Configuration Assistant, see *the first tutorial in this series*.)

There are three ways to catalog a database at a client:

- Automated configuration using discovery
- Automated configuration using access profiles
- Manual configuration

All of these methods are described in subsequent panels. From a client perspective, cataloging databases using a profile or discovery is the easiest way to do this. Manual configuration requires knowledge of the database location and characteristics in order to successfully run the command.

To use either of the automated configurations, the DBA must either generate profiles for his or her users or set up discovery services within the DB2 database. We won't go into the details of creating either of these facilities in this tutorial, but you're encouraged to to read the DB2 Administration manual for more details on these features.

Automated configuration using discovery

If you use this type of automated configuration, you do not need to provide any detailed communications information to enable the DB2 client to contact the DB2 server.

To add a database to your system using discovery, you'll need to walk through the following steps. (Note that a DB2 Administration Server (DAS) must be running and enabled for the discovery feature of the CA to return information about DB2 systems.)

1. Start the CA. You can do this from the Start menu on Windows, or via the `db2ca` command on either Windows or UNIX.

2. On the CA menu bar, under Selected, choose **Add Database Using Wizard**.

3. Select the Search the Network radio button and click **Next**.

4. Double-click on the folder beside Known Systems to list all the systems known to your client.

5. Click the plus sign (**+**) beside a system to get a list of the instances and databases on it. Select the database that you want to add, then click **Next**.

6. Enter a local database alias name in the Database Alias field. You can also enter a comment that describes this database in the Comment field, if you'd like.

7. If you are planning to use ODBC, register this database as an ODBC data source.
    ODBC must be already installed to perform this operation.

8. Click **Finish**.

You are now ready to use the database you added.

Automated configuration using access profiles

Access profiles are another automated method to configure a DB2 client to access
remote DB2 servers and their databases. An access profile contains the information
that a client needs to catalog databases to a DB2 server.

As with discovery, when using access profiles, you do not need to provide any detailed
communications information to enable the DB2 client to contact the DB2 server.

Two types of access profiles exist:

* *Server access profiles* are created from DB2 servers. They contain information about
  all the instances and databases the DB2 server has cataloged.

* *Client access profiles* are used for duplicating the cataloged databases and/or the
  client settings (DBM CFG, CLI/ODBC) from one client to another.

Both types of profiles can be exported from one DB2 system and then imported to
another.

You would typically use access profiles to configure a large number of clients. The DB2
Control Center can be used to export and then import a server access profile system.
(For more on the Control Center, see *the first tutorial in this series*.) A client access
profile is exported, then imported using the Configuration Assistant (CA).

If you have a large number of clients to configure, you should also consider making use
of LDAP (the Lightweight Directory Access Protocol). LDAP lets you store catalog
information in one centralized location. Each client just needs to know the centralized
location to be able to connect to any database that has been made available in the
network. See the DB2 Administration Guide for more details about LDAP.

Manual configuration

It is also possible to manually configure a database connection. To do this, you need to
know the details of the communications setup between the client and the server.

You can use a manual configuration to your host databases; use discovery to connect through a DB2 Connect server; or use that information for a direct connection from your client as described in the previous sections. There are two ways to manually configure connections:

- Use the Manual option in the CA. In this case, you are prompted via a GUI interface for all the values you need to enter.

- Use the CATALOG NODE/DB commands. In this case, you must know the syntax of the commands and enter the commands from a command-line interface.

In either case, you can use manual configuration to exploit some advanced options that are not available using automated methods -- you could choose the location where authentication takes place, for example.

The command-line CATALOG NODE/DB method is the trickier of the two, but it comes with an advantage: you can save the configuration steps into scripts so that the configuration can be redone if necessary.

Using the CA to catalog a database

In the following panels, you'll see the steps required to manually catalog a database using the Configuration Assistant.

Before proceeding, you need to know the following information:

- One of the protocols supported by the server instance containing the database
- The protocol connection information required to configure the connection to the server instance
- The server name
- The name of the database on the remote server

The initial CA screen is invoked by selecting the Configuration Assistant from the DB2 folder.

Command Line Tools	▶		
Development Tools	▶		
General Administration Tools	▶		
Information	▶		
Monitoring Tools	▶		
Set-up Tools	▶	Configuration Assistant	
		First Steps	
		Register Visual Studio Add-Ins	
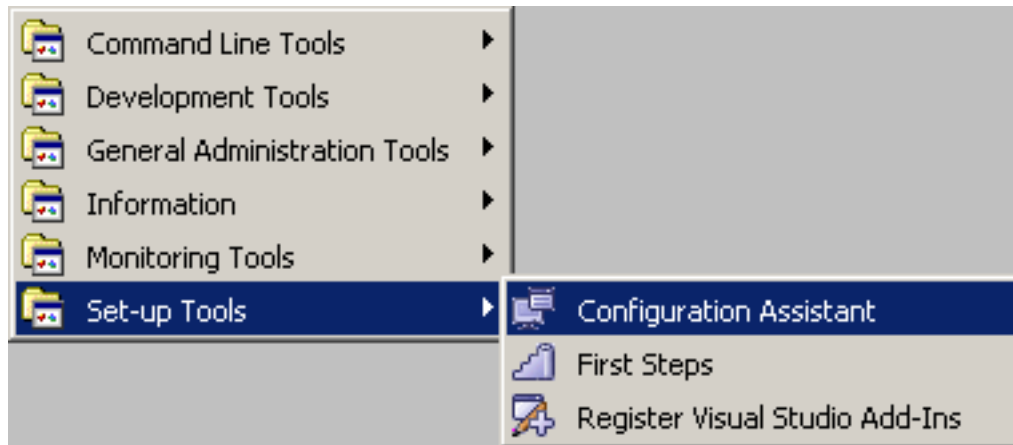
Once you've selected this program, the following screen will be displayed.

Configuration Assistant   _ □ X

Configure   Selected   Edit   View   Help

BAKLARZ - DB2

Alias	⬍	Name	⬍	Target Database	⬍	Location	⬍	Con
SAMPLE		SAMPLE				C:\DB2		
TOOLSDB		TOOLSDB				C:\DB2		

2 of 2 items displayed   ↓A/Z 🔁 ⬌ 🔍 ▷⁺ ▷⁻   Default ... ▲ View
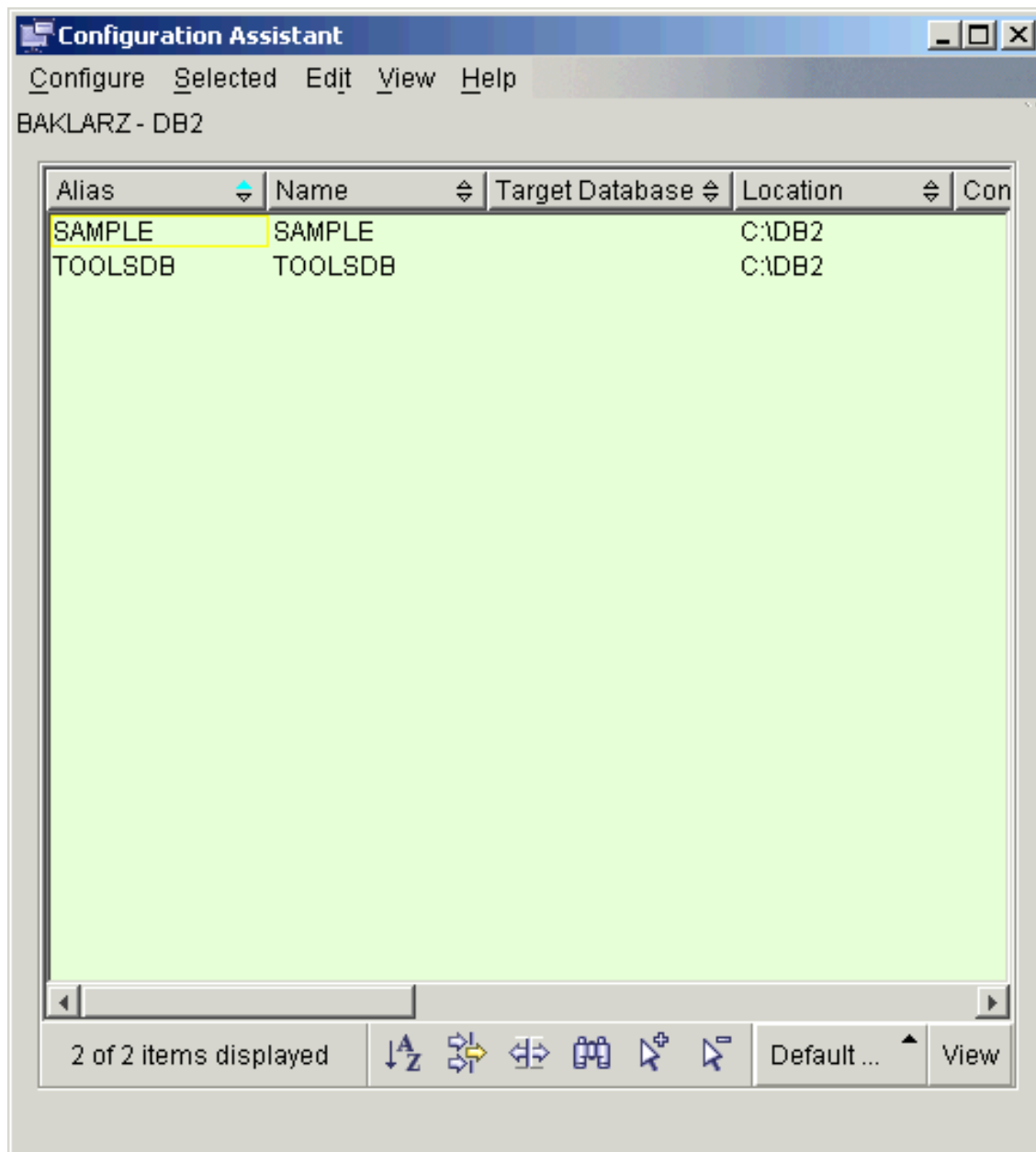
The top section of this screen gives the user a list of databases that are currently cataloged on the system. There are various menu items here that allow a user to configure the database connections, but the option that a DBA would be interested in is the Add Database Using Wizard button in the Selected menu:
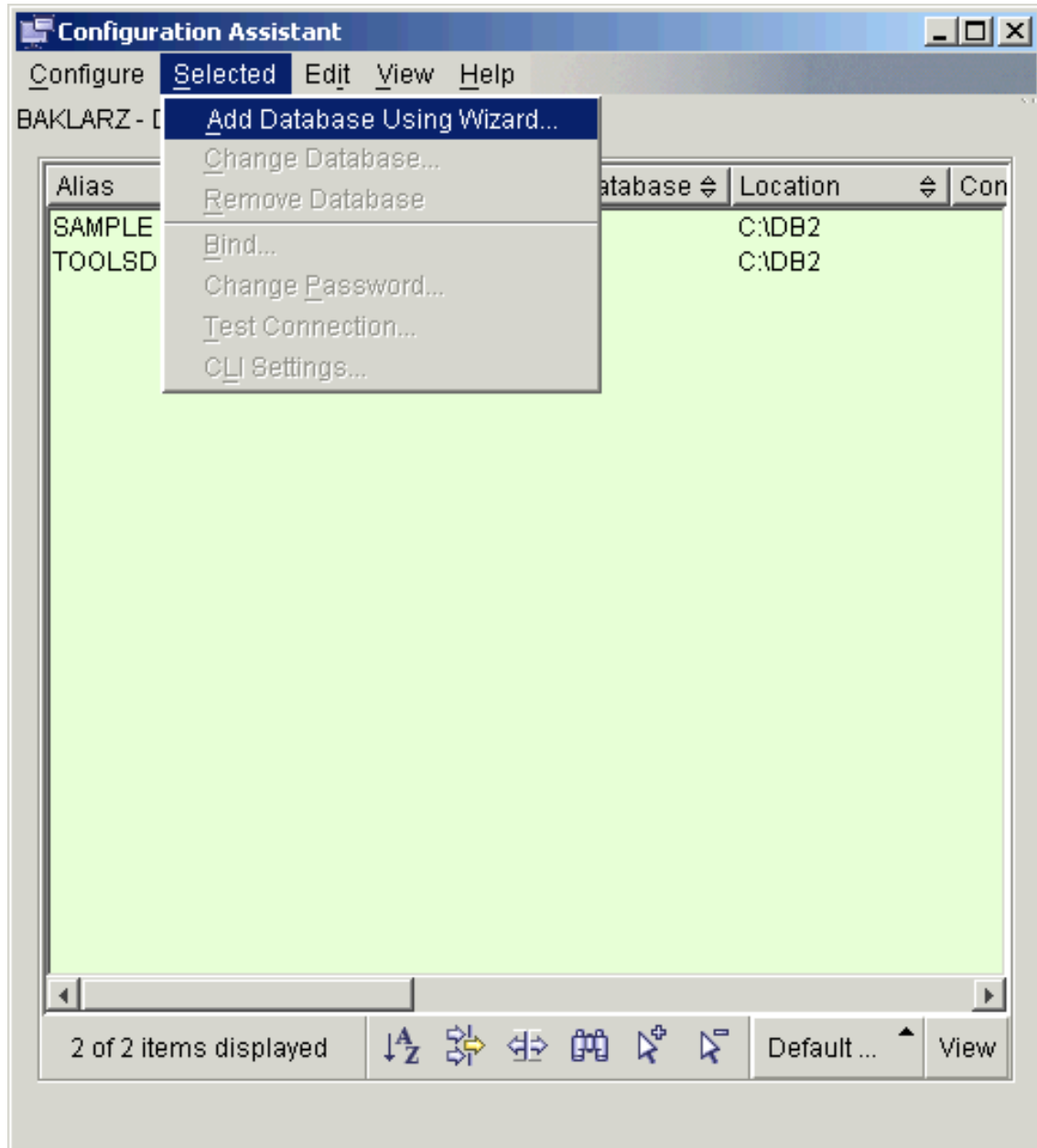
Choosing Add launches the Add Database Wizard.

The following panels will describe each step required in configuring the client.

CA: Which method to use?

The first panel of the CCA wizard asks you which method you would like to use to

catalog the database.
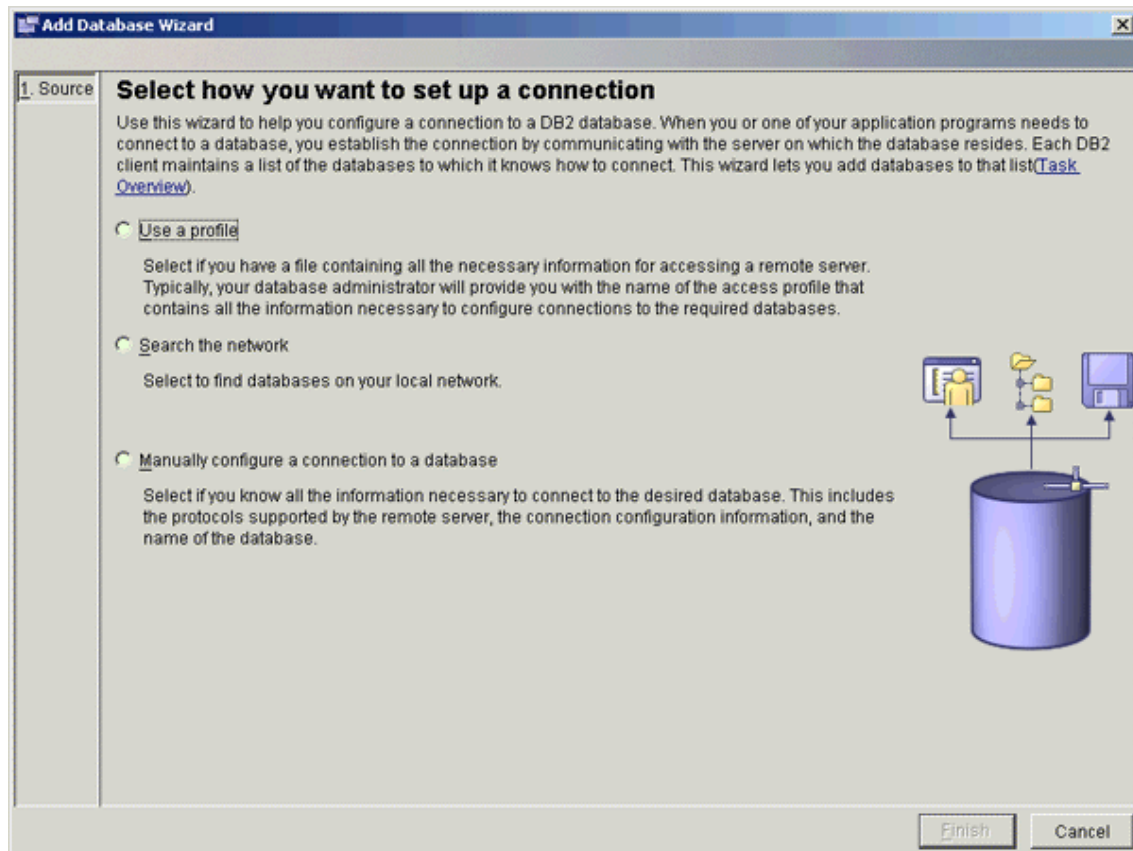
The wizard gives the user three possible ways of cataloging a database:

- Use a profile
- Search the network
- Manually configure a connection

The option that we are interested in is the manual configuration. (As the previous sections described, using a profile or searching the network are much easier methods for a user wanting to catalog a database on a client.)

Now that you've selected the manual configuration option, you need to select the communication protocol you want.

CA: Communication protocol

In the diagram below, you select the protocol that you will use to connect to the database. The contents of the Protocol Parameters box change according to the protocol. Here are the protocols you can choose, along with some of their parameters:

- **TCP/IP:** Server hostname/IP address, port number
- **NetBIOS:** Server workstation name, adapter number

- **Named Pipe:** Server computer name, instance
- **APPC/APPN:** Server symbolic destination name
- **LOCAL:** A local database on this machine

All of the protocols that DB2 supports are listed here, including APPC. If you have chosen APPC, your operating system choices will be: LAN-based, OS/390 or z/OS, OS/400, VM, or VSE. If you choose TCP/IP, your choices are: LAN-based, OS/390 or z/OS, OS/400, or VM.

You should check that the machine is properly configured on the network before clicking **Finish**.
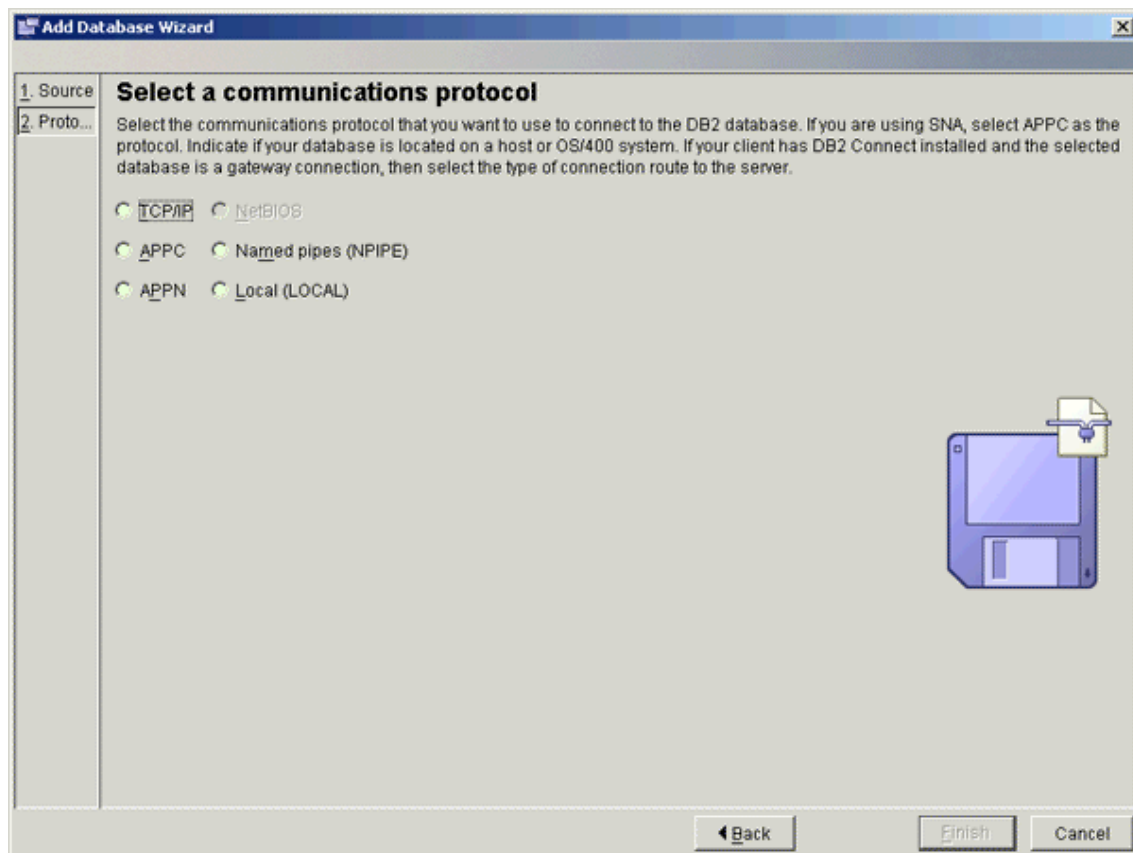
CA: Communication details

At this point, you get to enter the communication details of the database that you want to catalog. This screen will be different for each communication protocol. The example shown here is for a database that is found on the same machine as the client (local).

CA: Database details

Now you enter the details of the database that you want to catalog. Enter the name of the database (as known at the server) in the Database Name field. You can accept the same name as the local alias for the database, or change the alias to a name of your choice. You can also enter a description, if you'd like.

CA: ODBC settings

You can register the database as an ODBC data source. By default, the box indicating that you wish to do so is checked, as you can see in the figure below. You can optimize the ODBC settings for a particular application by selecting that application from the Optimize for Application menu.

If you plan to run Windows applications against this database, you should become familiar with the various optimization settings available in this environment.

CA: Node, system, and security options

Three additional panels are available, on which you can enter options relating to the remote node the database is on and the type of security being used.

1. **Node information:** You should fill in the node information if you plan to use the Control Center, since this will affect the behavior of this tool. The system and instance names are given by the values of DB2SYSTEM and DB2INSTANCE at the server. You should also select the operating system of the remote system.

2. **System information:** You need to supply the system information to tell DB2 the system, host, and operating system of the remote system on which the database resides.

3. **Security information:** Here you can specify where authentication of the user takes place (at the server, which is the default; at the client; or on a host or OS/400). You can also choose to use SOCKS security for TCP/IP connections, which would allow you to access a remote database outside your firewall. (For more on DB2 security, check out the *second tutorial in this series.*)
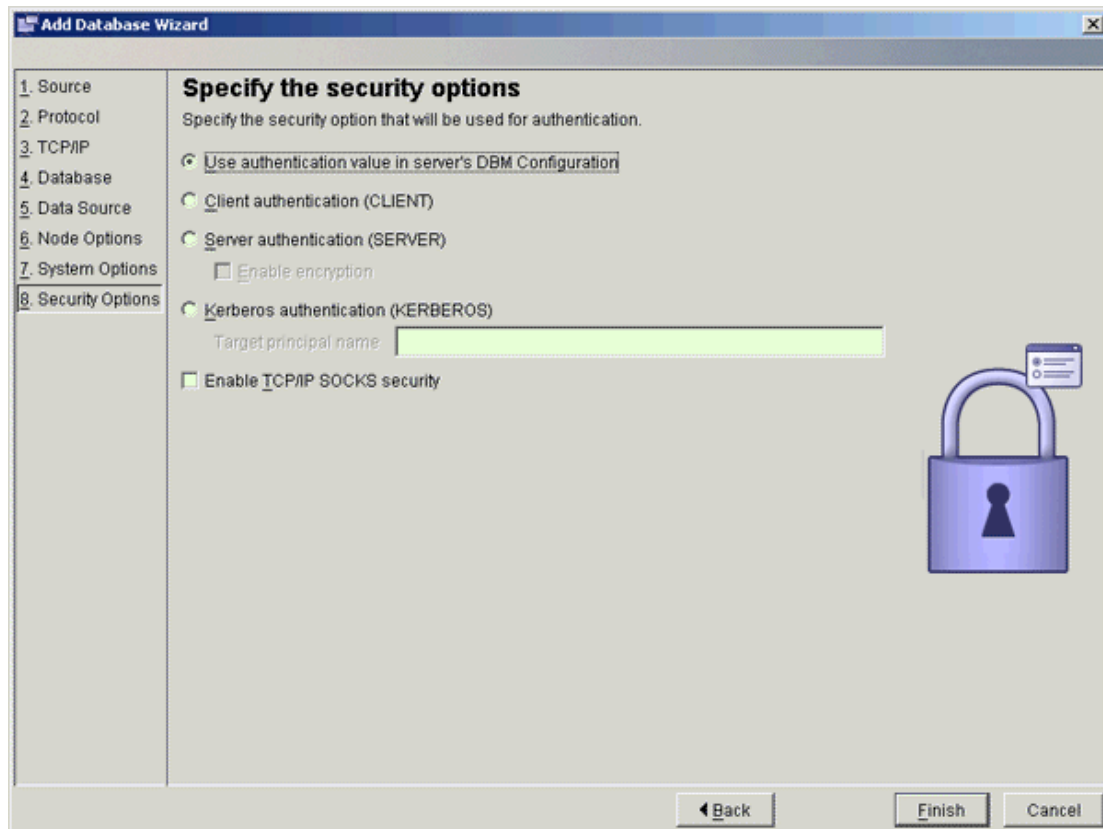
CA: Successful catalog operation

Once you click **Finish** on the CCA Catalog screen, DB2 will attempt to catalog the database. If this is successful, the following panel is displayed:
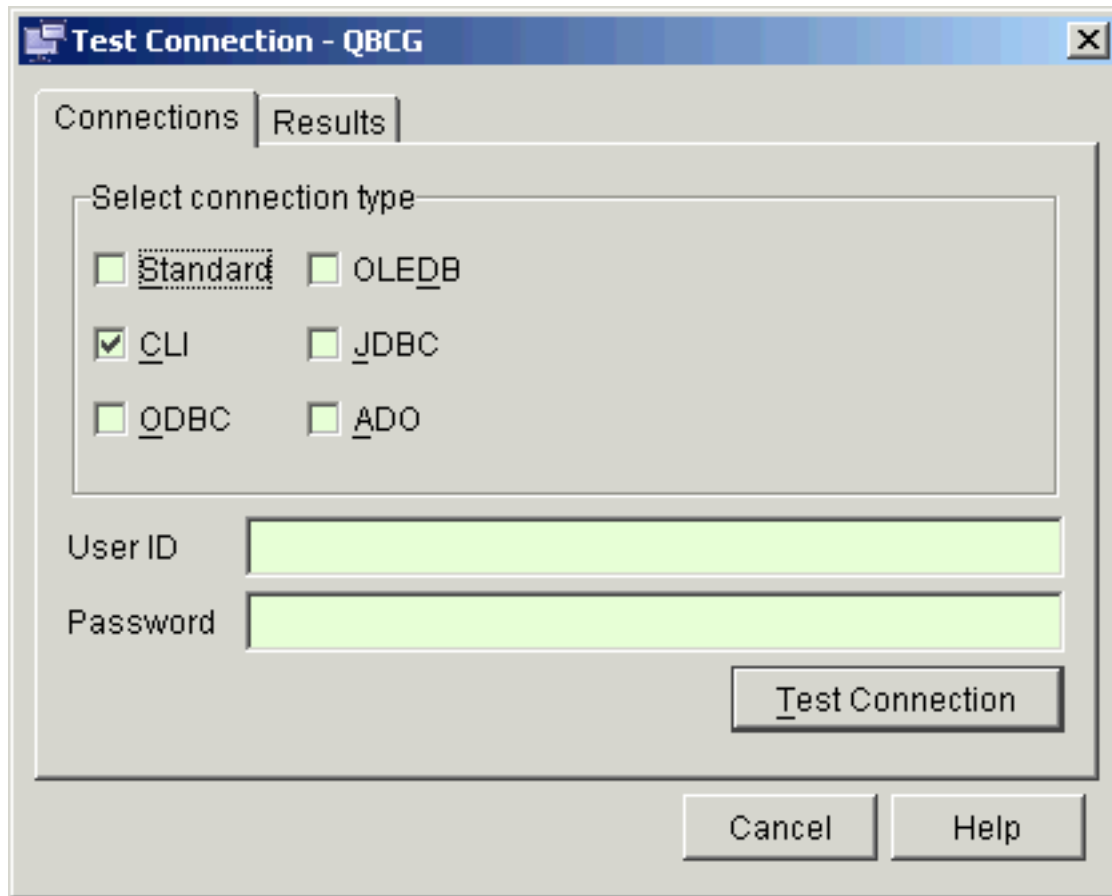
At this point, you can test the connection to make sure that the client can communicate with the database. If the test is successful, you can now use an application to access the DB2 database.

Summary

Cataloging a DB2 database is usually not required on the server where it was created. However, to access a database from a client, that client must first catalog the database locally so that applications can access it.

The CATALOG command can be used to catalog a database, but the Configuration Assistant (CA) is a much easier tool to use and allows for the automated discovery and cataloging of databases.

As an alternative to cataloging databases on every client, a DBA could also use LDAP services to create a central repository of database information.

Section 5. Manipulating DB2 objects with the Control Center

Using the Control Center

The *Control Center* is the central point of administration for DB2. The Control Center provides the user with the tools necessary to perform typical database administration tasks. It allows easy access to other server administration tools, gives a clear overview of the entire system, enables remote database management, and provides step-by-step assistance for complex tasks.

So why did we go through all that pain of learning how to create databases from the command line? Although the Control Center makes your life easier, there are times when you want to create scripts that automatically create objects or invoke database maintenance. The Control Center can help you generate, manage, and schedule these scripts, but they are all run as DB2 commands. And in some cases, the Control Center may not be available on the operating system you are using, so you'll have no alternative but to use DB2 commands.

The Control Center is invoked from within the General Administration folder in the DB2 program group:
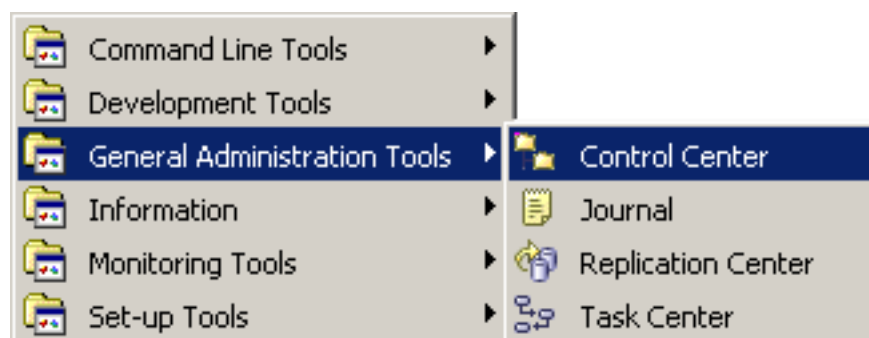
The screen that is displayed will be similar to the following figure:

Contents Pane Tool bar
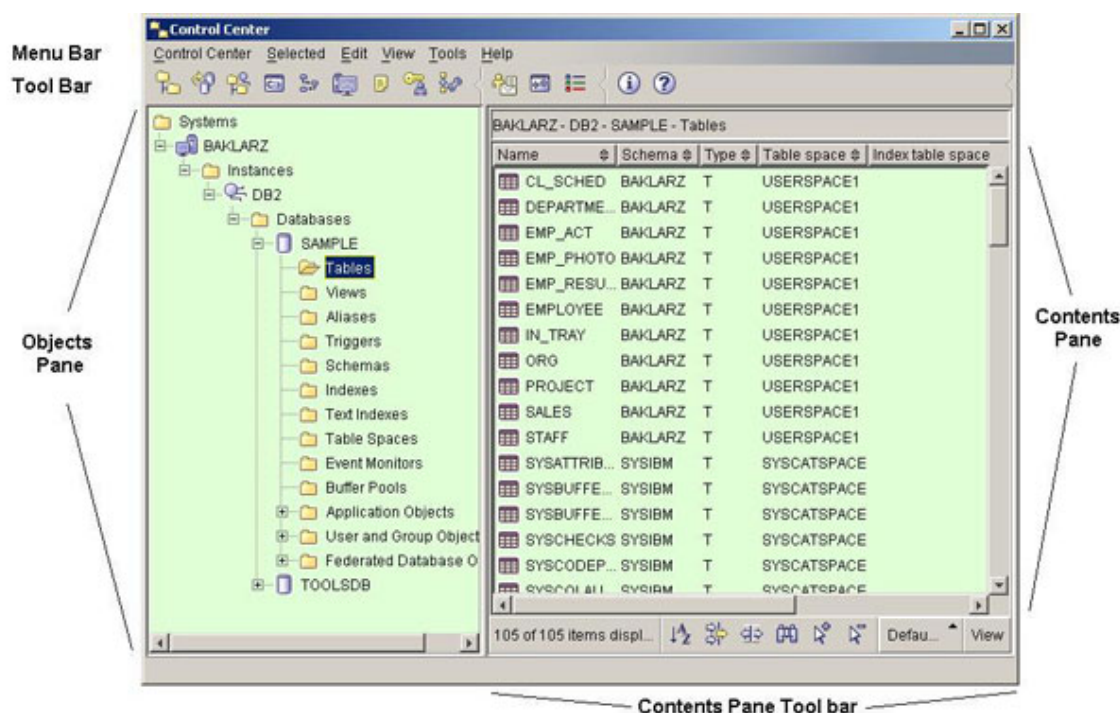
The Systems object represents both local and remote machines. To display all the DB2 systems that your system has cataloged, expand the object tree by clicking on the plus sign (**+**) next to Systems. The left portion of the screen lists available DB2 systems (local and remote). In the figure, the system LOCAL contains a DB2 instance, DB2, where the database SAMPLE is located. When Tables is highlighted, details about each system are shown in the Contents Pane. A number of the existing tables in the SAMPLE database are displayed in the figure above.

The main components of the Control Center are:

- **Menu Bar:** Used to access Control Center functions and online help.
- **Tool Bar:** Used to access the other administration tools.
- **Objects Pane:** Shown on the left-hand side of the Control Center window. It contains all the objects that can be managed from the Control Center as well as their relationship to one another.
- **Contents Pane:** This is found on the right side of the Control Center window and contains the objects that belong or correspond to the object selected on the Objects Pane.
- **Contents Pane Toolbar:** These icons are used to tailor the view of the objects and information in the Contents Pane. These functions can also be selected in the View menu.

Hover Help is also available in the Control Center, providing a short description for each icon on the toolbar as you move the mouse pointer over that icon.

Creating and manipulating databases

The Control Center can be used to create and manage your databases. Remember the CREATE DATABASE command you used earlier to create your first database? In the next few panels, you'll see how you would go about doing the same thing with the Control Center.

On the left side of the Control Center (the Objects Pane), place your mouse over the Database keyword and right-click on it. This will bring up a menu of the options that are available for databases. In this case, you would select **Create => Database Using Wizard**, as shown in the figure below.
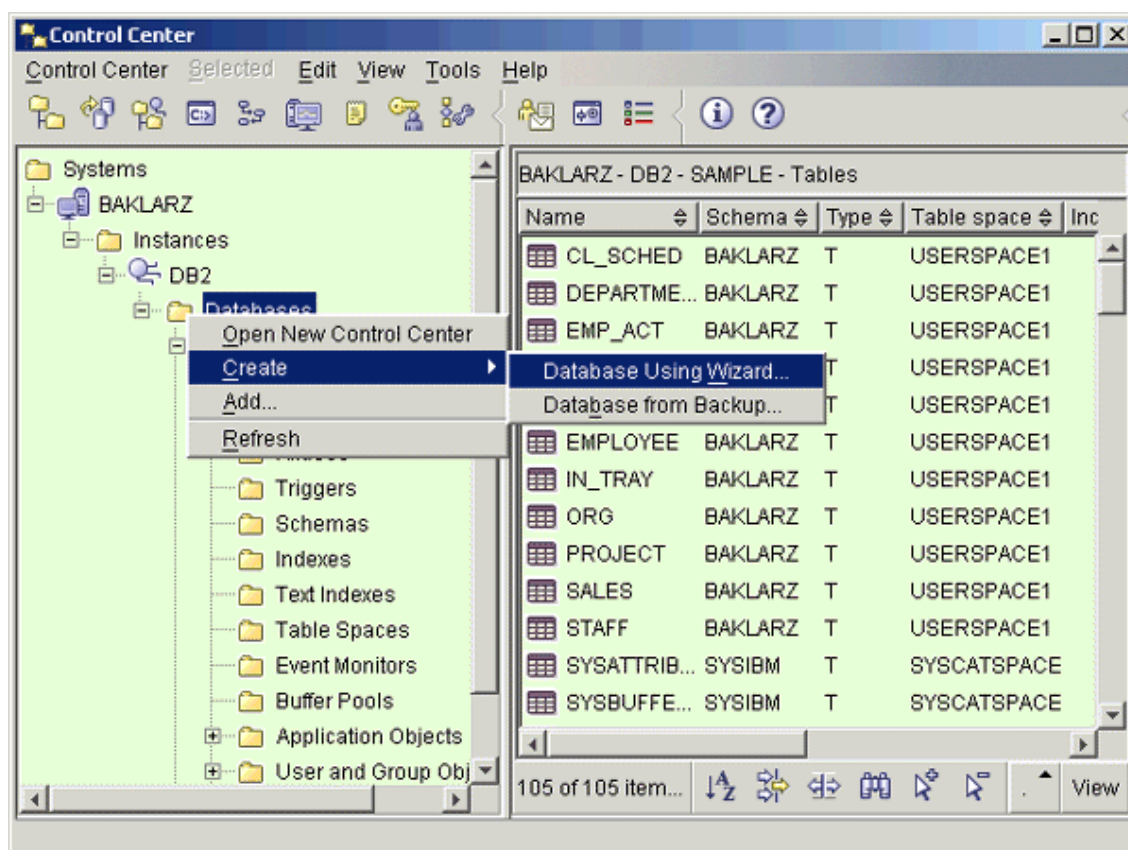
Once you've selected this option, DB2 will present a series of panels that you will need to fill in to create a database. The next series of panels will show how you can use these wizards to simplify some of the common DBA tasks.


Create Database wizard: Database name

The Create Database Wizard will prompt you through a number of steps to generate a database. The first screen asks for the name of the database, the default drive where you want it created (if you don't specify anything else), and an alias name. In addition, you can add a comment about contents of the database.
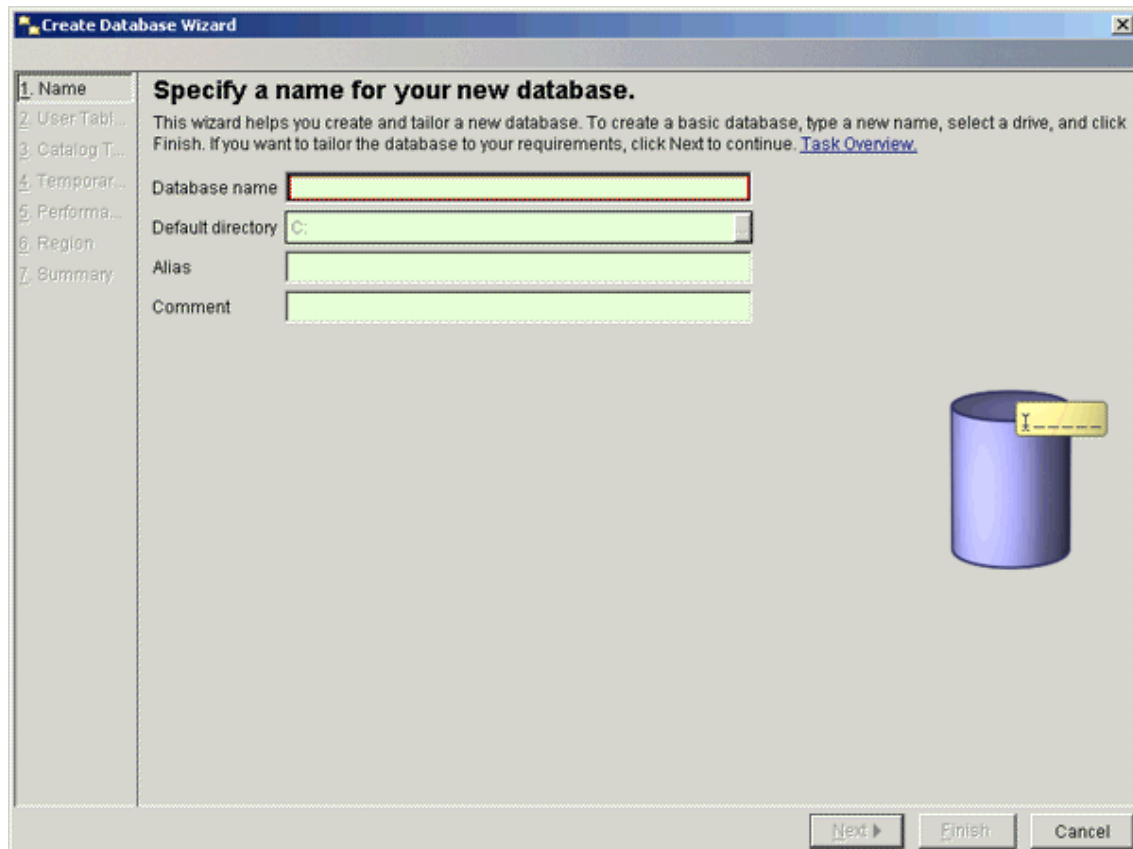

Create Database Wizard: User/catalog/temp tables

The next three panels of the wizard ask you to fill in information on how you would like the user, catalog, and temporary tablespace to be created. If you select the Low Maintenance option, the wizard will create an SMS tablespace for you. If you select High Performance, you'll need to specify the devices and file systems that you plan to use for this tablespace.
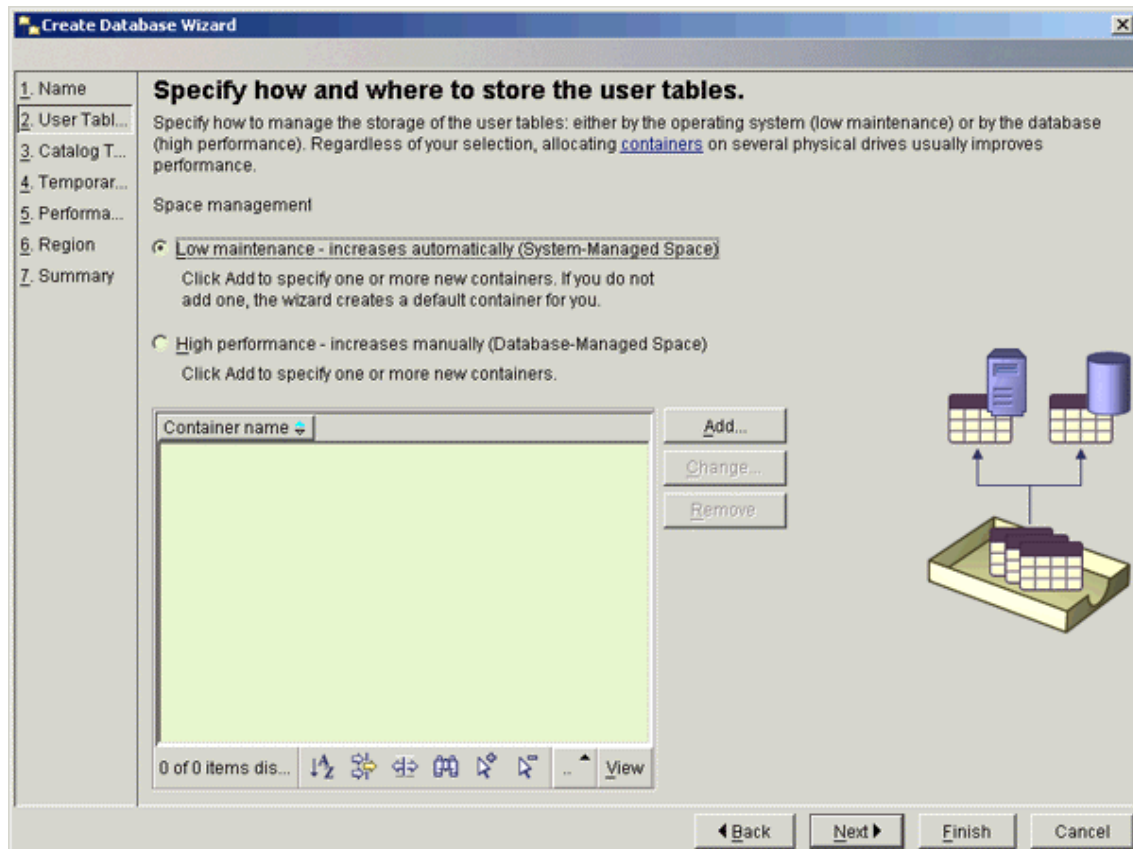
For either option, you can specify the containers (files, devices) that you want to allocate to the tablespace. If you click the Add button, an additional panel will be displayed to allow you to define the containers being used.
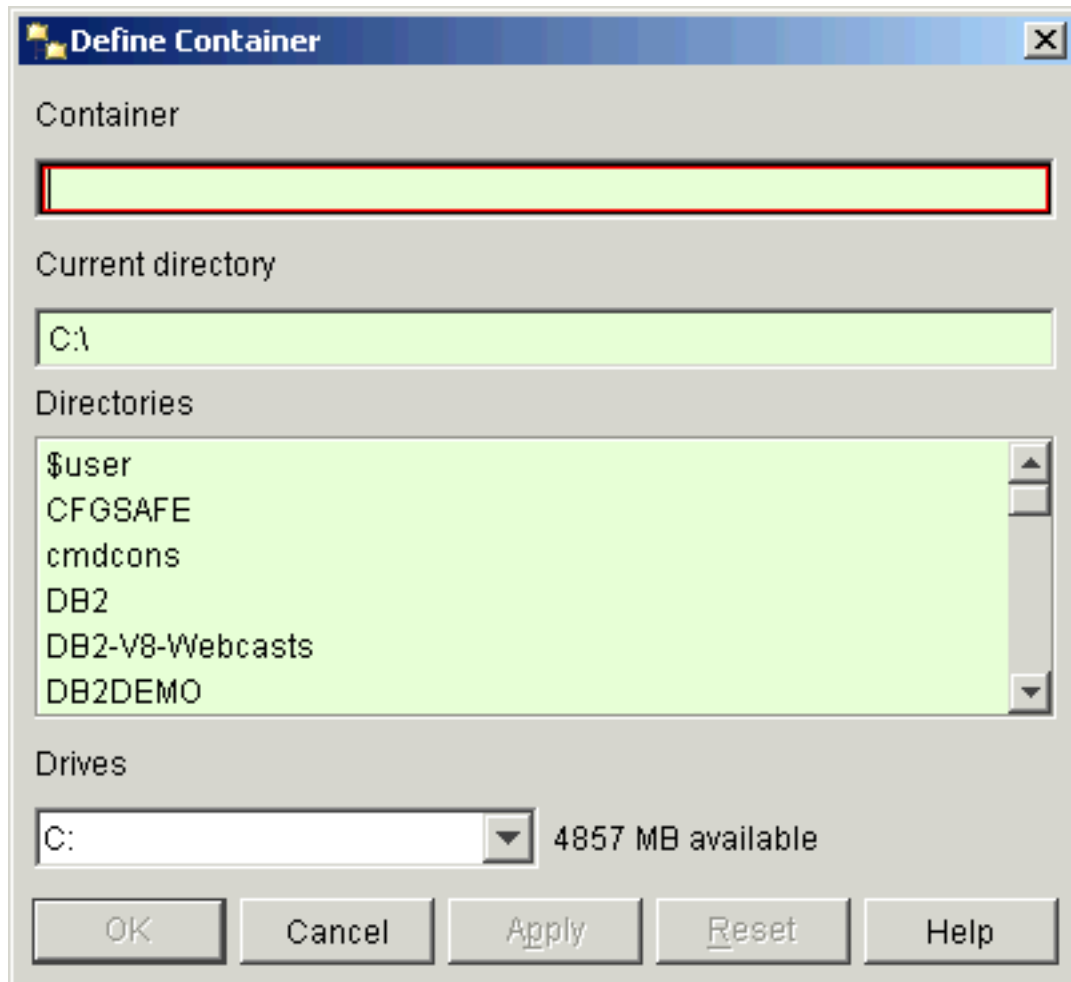
If you do not specify containers or files for your tablespaces, DB2 will automatically generate one for you on the default drive that you specified earlier.

Create Database Wizard: Performance options

There are two performance parameters that you can set: EXTENTSIZE and PREFETCHSIZE.
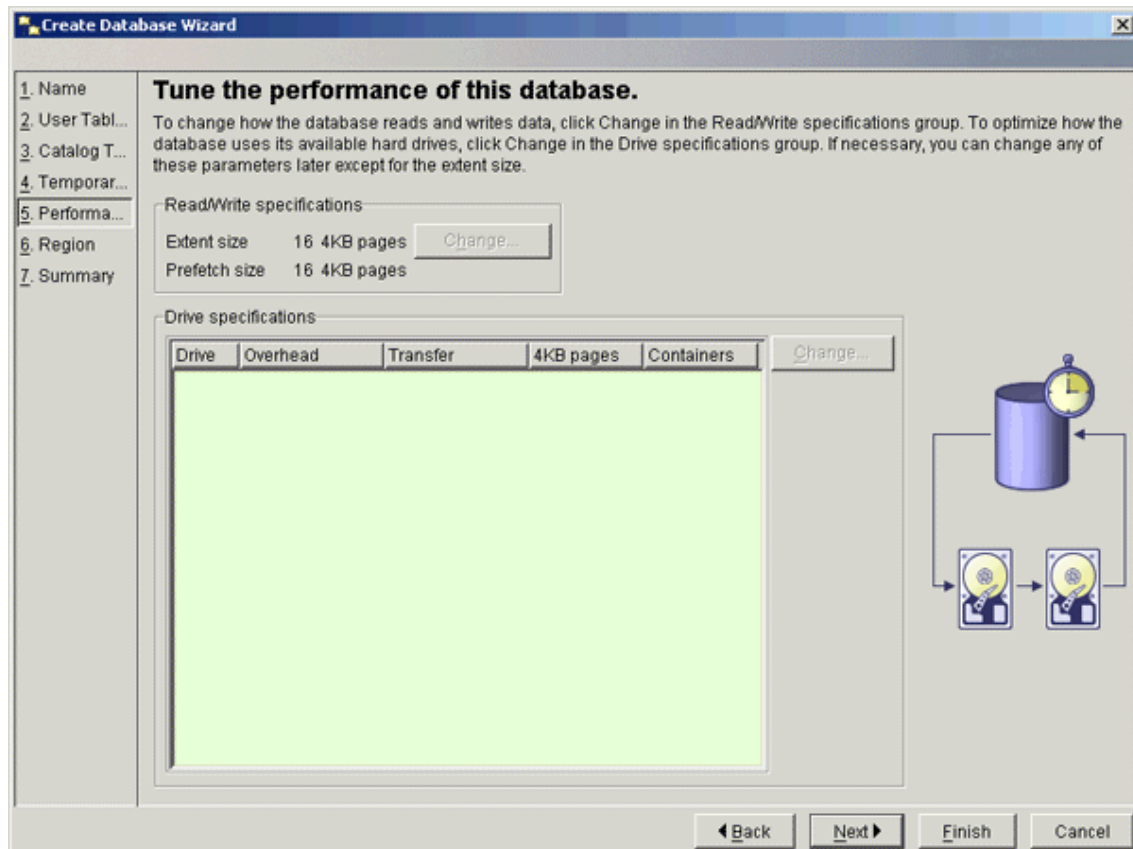
Let's look at each of them:

- EXTENTSIZE: An *extent* is a unit of space within a container of a tablespace. Database objects (except for LOBs and long varchars) are stored in *pages* within DB2. These pages are grouped into extents. The extent size is defined at the tablespace level. Once the extent size is established for the tablespace, it cannot be altered. The database configuration parameter DFT_EXTENT_SZ specifies the default extent size for all tablespaces in the database. This value can range from 2 to 256 pages; thus, the absolute size would range from 8 KB to 1024 KB for 4 KB pages, or from 16 KB to 2048 KB for 8 KB pages. This figure can be overridden by using the EXTENTSIZE parameter in the CREATE TABLESPACE statement.

  If you intend to use multidimensional clustering (MDC) in the design of your tables, the extent size will become a critical design decision. MDC tables will allocate an extent for each new dimension set that is created. If the extent size is too large, then there is a possibility that much of the extent will be empty (for dimension sets with few records). For more information on MDC and its impact on EXTENTSIZE, refer to the DB2 Administration Guide.

- PREFETCHSIZE: Sequential prefetching is the ability of the database manager to anticipate a query in advance, reading pages before those pages are actually referenced. This asynchronous retrieval can reduce execution times significantly. You can control how aggressively the prefetching is performed by changing the PREFETCHSIZE parameter on the CREATE TABLESPACE statement. By default this value is set to the DFT_PREFETCH_SZ database configuration parameter. This value represents how many pages will be read at a time when a prefetch request is triggered by DB2. By setting this value to a multiple of the extent size, multiple

extents can be read in parallel. This function is even more effective when the containers for the tablespace are on separate hard disks.

The default values for these parameters are adequate for many applications, but you may want to experiment with higher PREFETCHSIZEs for applications that do heavy queries or analyze large amounts of data.

Create Database Wizard: Code page and collating sequence

The next option you'll encounter as part of database creation are those involving code pages and collating sequences.
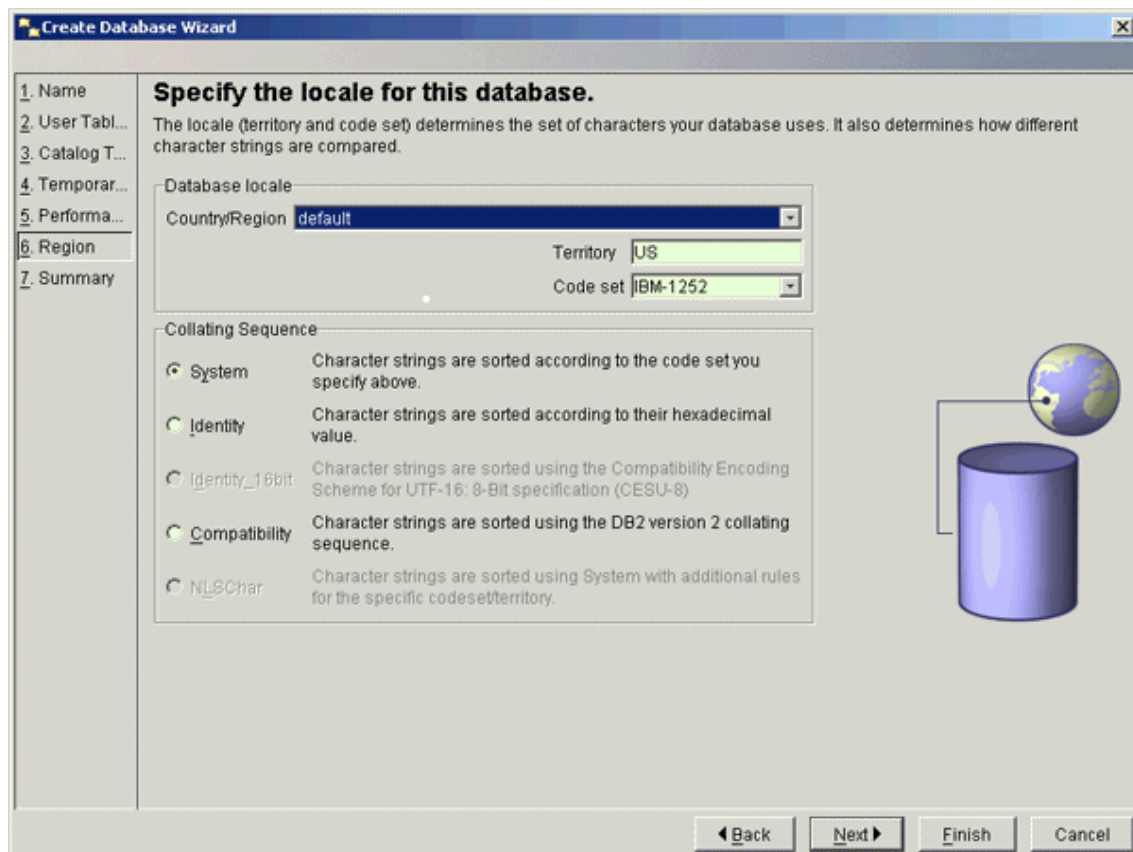

When a DB2 application is bound to a DB2 database, the application and database code page are compared. If the code pages are not equal, code page conversion will be attempted for each SQL statement. If you are using a code page other than that of the database you are accessing, it is important to ensure that the code pages are compatible and conversion can be accomplished.

By default, the collating sequence of a database is defined according to the codeset used in the CREATE DATABASE command. If you specify the option COLLATE USING SYSTEM, the data values are compared based on the TERRITORY specified for
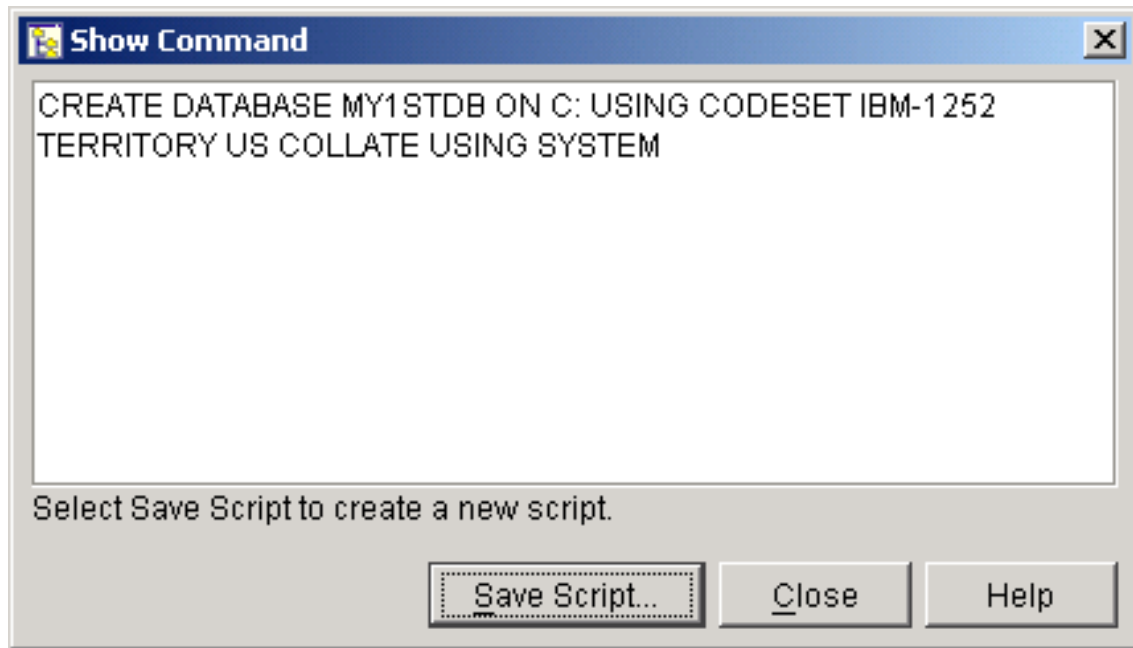
the database. If the option COLLATE USING IDENTITY is used, all values are compared using their binary representation in a byte-to-byte manner. When you need to store data in its native (binary) format, avoid using data types with code pages. It is generally advantageous to use identical application and database code pages to avoid the code page conversion process.

Create Database Wizard: Create summary

Once all of your parameters have been entered into the system, the Create Database Wizard will present you a summary screen with all of the selections that you have made.
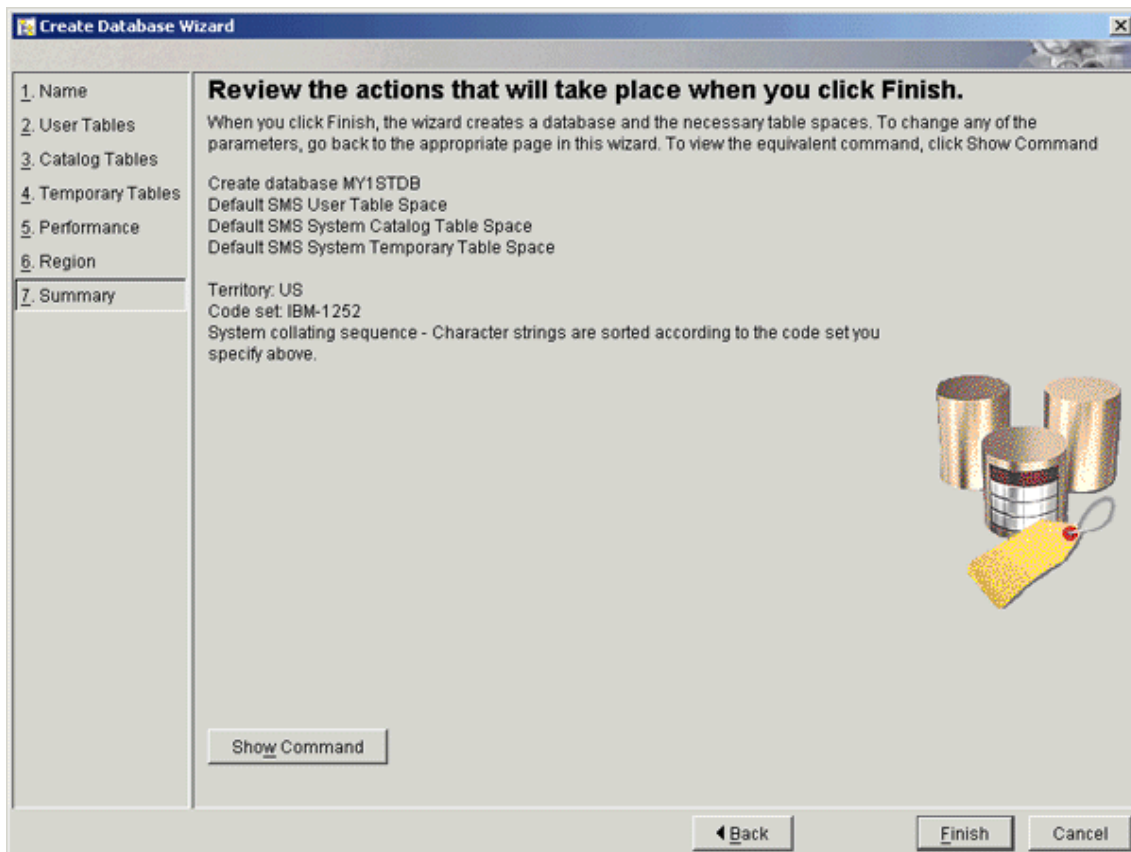
One extremely useful feature of this summary page is its Show Command button. If you press it, you'll see the DB2 command that will be used to create the database, as shown in the figure below.

```
Show Command                                                                      ⊠

CREATE DATABASE MY1STDB ON C: USING CODESET IBM-1252
TERRITORY US COLLATE USING SYSTEM




Select Save Script to create a new script.

                              [ Save Script... ]      [ Close ]        [ Help ]
```

You can save this command for later execution, or cut and paste it into a script that you
might be developing. If you are satisfied with the parameters that you have entered into
the system, click the Finish button to create the database.

Summary

The DB2 Control Center is a powerful tool that makes routine database maintenance
extremely easy. The are a variety of wizards available to help you create or modify
many of the database objects.

You can also use the Control Center to generate DB2 commands for later use in scripts
or programs. This feature allows you to develop the command you want to use without
actually executing it.

Section 6. Creating and accessing basic DB2 objects

More Control Center tricks

While the Control Center is certainly useful for creating databases, it has a lot of additional functionality you can use to create, modify, or delete almost any database object. Let's take a look at what else the control center has in it.
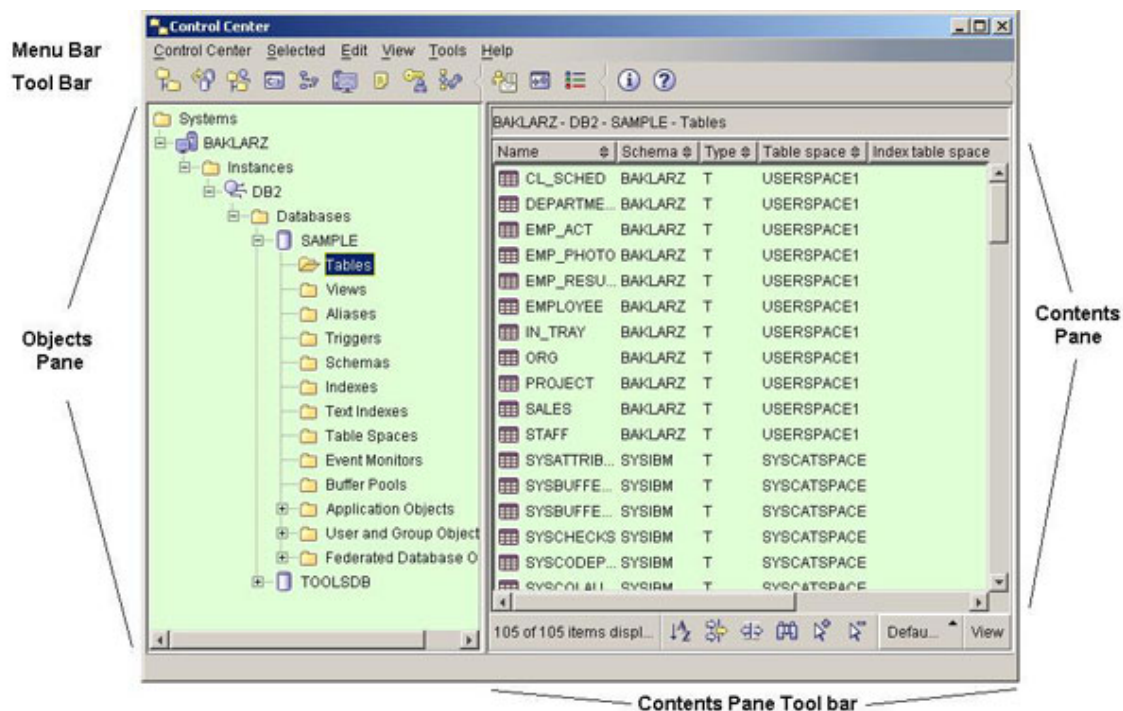

On the left size of the screen we have the Objects Pane. If you want to create a new object, place your mouse on the object type (table) and right-click it. For most objects, this will present a menu of options, including one for creating the object.
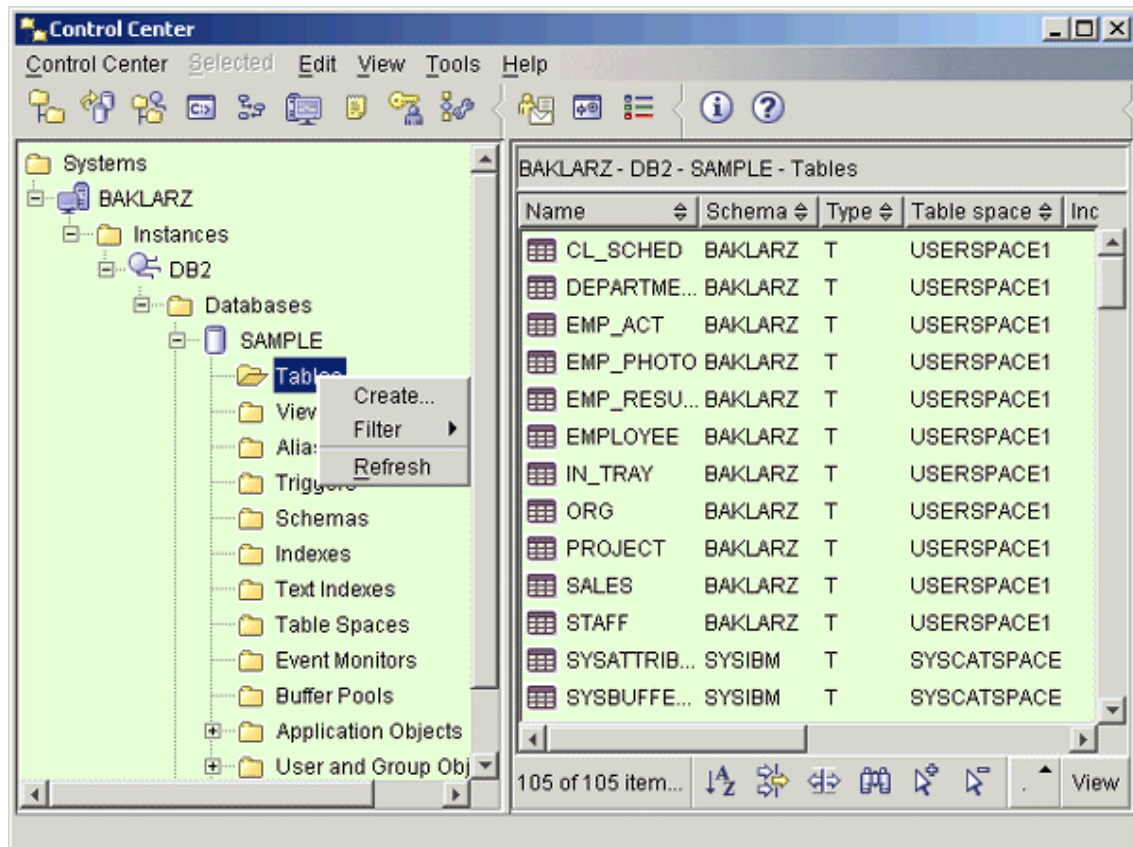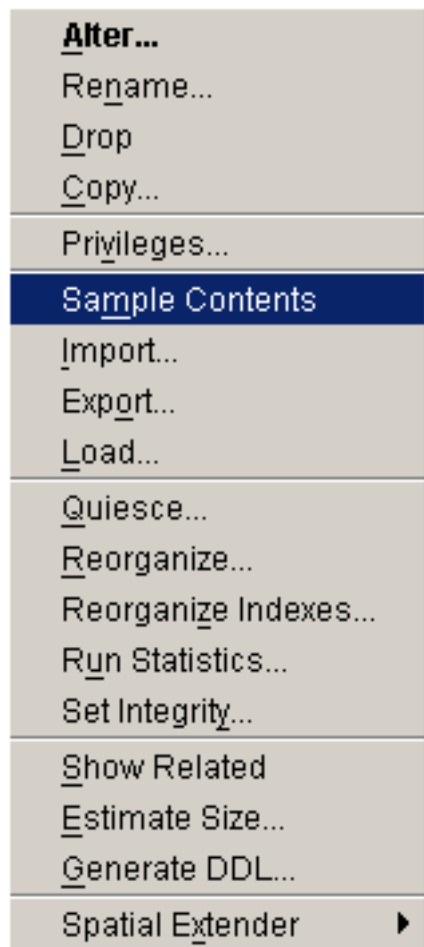
As a DBA, you should become familiar with these Create buttons. Initially, these wizards can be extremely useful in determining how the DB2 commands are generated. Taking advantage of the Show Command button can be a great learning tool. Even experienced DBAs aren't ashamed to use the Control Center to generate seldom-used commands!

Modifying existing objects

New objects are created by right-clicking on the object name in the Objects Panes. If you need to modify or delete an object, you need to display it in the Contents Pane on the right side control center.
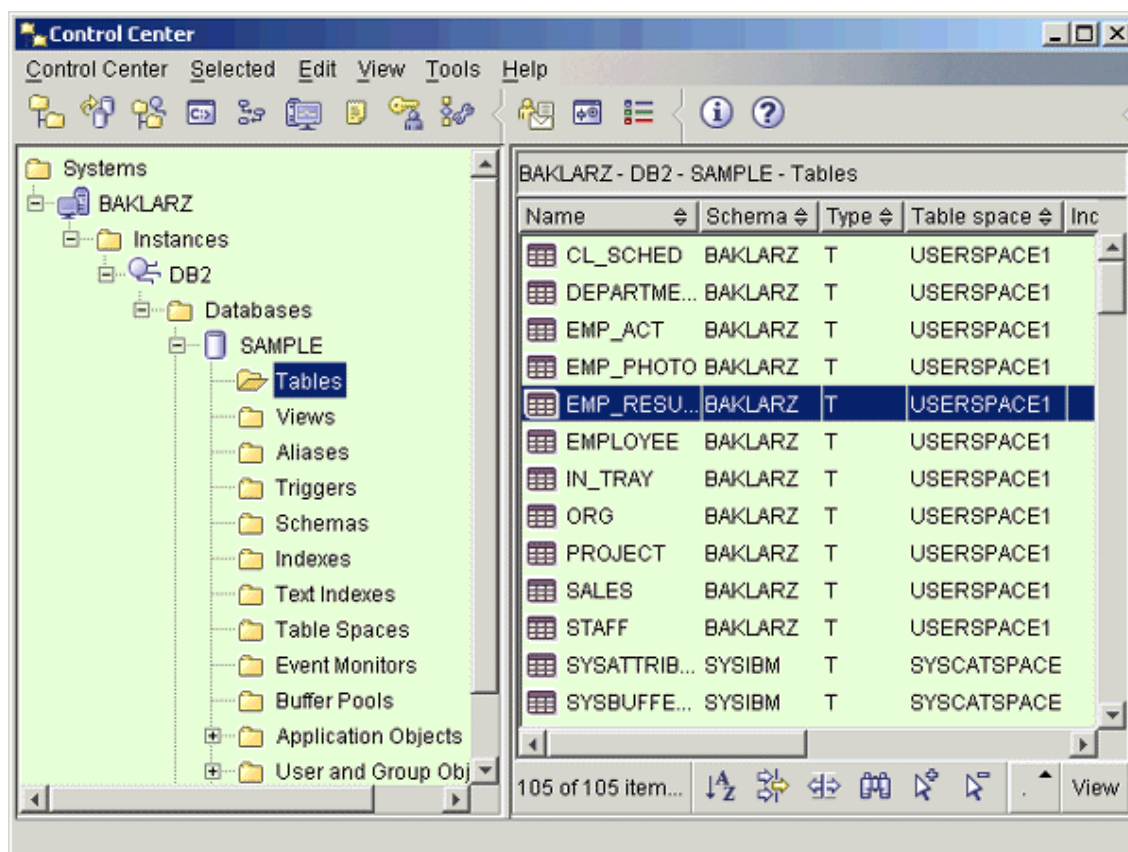
In the panel shown in the figure above, the tables found in the SAMPLE database are listed in the Contents Pane. Now that these tables are displayed, you can modify them. Right-click on the object you want to modify. You'll see the following menu:

Alter...
Rename...
Drop
Copy...
Privileges...
Sample Contents
Import...
Export...
Load...
Quiesce...
Reorganize...
Reorganize Indexes...
Run Statistics...
Set Integrity...
Show Related
Estimate Size...
Generate DDL...
Spatial Extender   ▶

There are a number of actions that can be performed against table objects. For example, you can view the sample contents of the table, alter the table, or reorganize it.

Two commands that are of interest to DBAs are the Show Related and Generate DDL commands. Show Related will display all database objects, like indexes or tablespaces, that are related to this table. The Generate DDL command will reverse-engineer the definition of this table, so that you can re-create the table design in the event of a failure. (Of course, you always back up your database, correct?)

Section 7. Wrapup

Summary

The DB2 Control Center can also be used to modify objects that exist within the database. For each object type, you are presented with various options defining what they can change.

Most actions that a DBA takes from within the Control Center can be captured and saved for use in a script. This, along with the command wizards, can make maintenance on the database much simpler than typing in commands.

Additional information on the DB2 Control Center can be found in the online help supplied with the tool. In addition, the DB2 Administration Guide and the DB2 Command Reference hold a wealth of information on database features, functions and how to design a database for the best performance. These books are an excellent reference and should be kept close at hand when designing your databases!

Resources

For more information on the DB2 Fundamentals Exam 700:

- *IBM Data Management Skills information*
- Download a *self-study course for experienced Database Administrators (DBAs)* to quickly and easily gain skills in DB2 UDB.
- Download a *self study course for experienced relational database programmers* who'd like to know more about DB2.
- *General Certification information* -- including some book suggestions, exam objectives, courses

Check out the other parts of the DB2 V8.1 Family Fundamentals Certification Prep series:

- *DB2 V8.1 Family Fundamentals Certification Prep, Part 1 of 6: DB2 Planning*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 2 of 6: DB2 Security*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 4 of 6: Working with DB2 UDB Data*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 5 of 6: Working with DB2 UDB Objects*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 6 of 6: Data Concurrency*

Feedback

Colophon

This tutorial was written entirely in XML, using the developerWorks Toot-O-Matic tutorial generator. The open source Toot-O-Matic tool is an XSLT style sheet and several XSLT extension functions that convert an XML file into a number of HTML pages, a zip file, JPEG heading graphics, and two PDF files. Our ability to generate multiple text and binary formats from a single source file illustrates the power and flexibility of XML. (It also saves our production team a great deal of time and effort.)

You can get the source code for the Toot-O-Matic at www6.software.ibm.com/dl/devworks/dw-tootomatic-p. The tutorial Building tutorials with the Toot-O-Matic demonstrates how to use the Toot-O-Matic to create your own tutorials. developerWorks also hosts a forum devoted to the Toot-O-Matic; it's available at www-105.ibm.com/developerworks/xml_df.nsf/AllViewTemplate?OpenForm&RestrictToCategory=11. We'd love to know what you think about the tool.